BOJ 1915 풀이

가장 큰 정사각형

Approach

2573 빙산 ?

[0][0]~[N][M] 돌면서
빙산이 나오면 덩어리 체크하고~~
녹이고~~ 등등

➔ **인접한 빙산을 봐가면서**
DFS로 풀었던 문제 !

빙산 출처 분류

시간 제한	메모리 제한	제출	정답	맞은 사람	정답 비율
1 초	256 MB	31365	8852	5872	26.098%

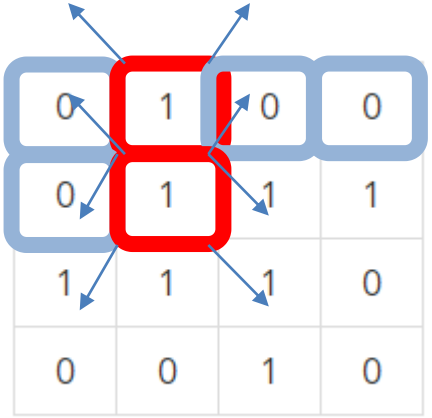
문제

지구 온난화로 인하여 북극의 빙산이 녹고 있다. 빙산을 그림 1과 같이 2차원 배열에 표시한다고 하자. 빙산의 각 부분별 높이 정보는 배열의 각 칸에 양의 정수로 저장된다. 빙산 이외의 바다에 해당되는 칸에는 0이 저장된다. 그림 1에서 빈칸은 모두 0으로 채워져 있다고 생각한다.

	2	4	5	3		
	3		2	5	2	
	7	6	2	4		

필요한 과정

1. 1이 나오면 대각선으로 확장 가능한 지 체크
2. 가능 ➔ 확장한 넓이와 max와 비교

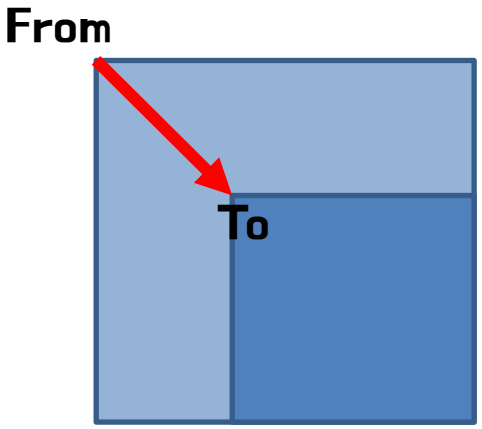


CODE

isSquare()
→확장 했을 때 정사각형인지

```
public static boolean isSquare(int row, int col, int newR, int newC) {

    int rFrom=(row<newR)?row:newR; //row와 newR중 작은게 From
    int rTo=(row>newR)?row:newR;
    int cFrom=(col<newC)?col:newC;
    int cTo=(col>newC)?col:newC;
```



From →To : for문을 돌며
연한 파란색 부분에 0이 있는지 체크

isSquare()

```
boolean visitedTmp[][]=new boolean[n][m]; //정사각형이 아닐수도 있으니 복사본을 수정해가며 처리
for(int i=0;i<n;i++) {
    for(int j=0;j<m;j++) {
        visitedTmp[i][j]=visited[i][j];
    }
}

for(int i=0;i<=rTo-rFrom;i++) {
    for(int j=0;j<=cTo-cFrom;j++) {
        if(map[rFrom+i][cFrom+j]==0) {
            area=1;
            return false;
        }
        visitedTmp[rFrom+i][cFrom+j]=true
        area++;
    }
}

for(int i=0;i<n;i++) {
    for(int j=0;j<m;j++) {
        visited[i][j]=visitedTmp[i][j];
    }
}
area-=1; //연결지점은 중복되므로 빼주기
max=Math.max(max, area);
return true;
```

➔ visitedTmp에 수정 현황을 저장

➔ From에서 to로 돌며 0이 나오면 return, area복구

1이면 area++

➔ From에서 to로 돌았는데 모두 1이면

Area에서 연결지점 빼 줌 최대크기와 비교

DFS()

```java
public static void DFS(int row, int col) {
    for(int i=0;i<pos.length;i++) {
        int newR = row + pos[i][0];
        int newC = col + pos[i][1];

        if(newR>=0&&newR<n && newC>=0&&newC<m && !visited[newR][newC]) { //배열 범위 체크
            if(isSquare(row,col,newR,newC)) { //대각선으로 키워도 정사각형이라면
                DFS(newR,newC);
            }
        }
    }
}
```

```java
static int pos[][]= {{-1,-1},{-1,1},{1,-1},{1,1}}; //상+좌,상+우,하+좌,하+우
```

1. 4방향의 대각선을 좌표로 쉽게 이용 하고자
미리 생성한 pos를 더해준다.

2. 대각선으로 이동한 좌표가 (범위내 && 정사각형 유지)라면
또 DFS를 호출해서 쭉쭉 확장

main()

```
for(int i=0;i<n;i++) {
    String str=br.readLine();
    for(int j=0;j<m;j++) {
        String tmp=str.charAt(j)+"";
        map[i][j]=Integer.parseInt(tmp);
        if(map[i][j]==0) visited[i][j]=true; //0은 미리 방문처리해서 체크안하도록
    }
}

for(int i=0;i<n;i++) {
    for(int j=0;j<m;j++) {
        if(!visited[i][j]) {
            area=1;
            DFS(i,j);
        }
    }
}
```

➔ 입력 받음.
0은 애당초 방문처리

➔ Map을 돌며 1이 나오면
DFS